Prevent, Mitigate, and Recover (PMR) Insight Collective Knowledge System (PICK)

CRC Document

Version 1.0

2/21/2020

Document Control

Approval

The Guidance Team and the customers shall approve this document.

Document Change Control

|  |  |
| --- | --- |
| Initial Release: | 1.0 |
| Current Release: | 1.1 |
| Indicator of Last Page in Document: | $ |
| Date of Last Review: | 2/21/2020 |
| Date of Next Review: | 2/24/2020 |
| Target Date for Next Update: | 2/24/2020 |

Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:

Steven Roach

Customer: Dr. Oscar Perez

Vincent Fonseca

Herandy Denisse Vazquez

Baltazar Santaella

Florencia Larsen

Erick De Nava

Software Team Members:

Mark Williams

Valente Arellano

Joseph Warren

TJ Haddad

Ivan Torres

Change Summary

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 1.0 | 2/18/2020 | All team members | Completed and merged CRC cards for all classes |
| 1.1 | 2/21/2020 | Mark Williams | Updated document formatting |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[Document Control ii](#_Toc33173323)

[Approval ii](#_Toc33173324)

[Document Change Control ii](#_Toc33173325)

[Distribution List ii](#_Toc33173326)

[Change Summary ii](#_Toc33173327)

[1. Introduction 1](#_Toc33173328)

[2. Collaborations 1](#_Toc33173329)

# Introduction

The purpose of the CRC document is to brief the customers on the classes, responsibilities, and collaborations necessary for the construction of the PICK - the Prevent, Mitigate, and Recover (PMR) Insight Collective Knowledge System (PICK) – tool . Essentially, the PICK tool aims to provide the Lethality, Survivability and HSI directorate (LSH) with the capability of correlating and graphically representing events that have occurred in an Adversarial Assessment (AA).

The CRC document defines the following:

* Classes: Collections of objects that will make up the system.
* Responsibilities: The knowledge or actions of a class
* Collaborations:

The intended audience of the SCM Plan is Dr. Oscar Perez, Mr. Vincent Fonseca, Ms. Herandy Vazquez, Mr. Baltazar Santaella, Ms. Florencia Larsen, Mr. Erick De Nava and the software development team.

# Classes

In this section, we describe all of the *classes* of the PICK tool. They are as follows:

* Event Configuration
* Log File
* Log File Manager
* Enforcement Action Report
* Enforcement Action Report Manager
* Vector
* Log Entry
* Node
* Icon
* Graph
* Audio Log File
* Video Log File
* Relationship
* Vector Database
* Lead Vector Database
* User Vector Database
* Image Log File
* PDF Log File
* Log Entry Manager
* Icon Manager
* Client Handler
* Server Handler
* Event Configuration Window
* Vector Configuration
* Log File Configuration
* Log Entry Configuration
* Vector DB Configuration
* Graph Configuration
* Icon Configuration
* Vector Table Configuration

# Responsibilities

In this section, we describe all of the *responsibilities* the classes of the PICK tool must have. They are as follows:

* Knows the name of the event
* Knows the start timestamp of an event
* Knows the end timestamp of an event
* Knows the path to the root directory of an event
* Knows the name of the red team folder
* Knows the name of the blue team folder
* Knows the name of the white team folder
* Knows whether or not the current user is the lead
* Knows the IP address of the lead
* Knows the number of connections established to the lead
* Can store all event configuration information in a system datastore
* Can validate directory structure
* Can validate directory existence
* Can perform data cleansing
* Can perform data ingestion
* Knows the file name of the log file
* Knows the date and time it was most recently updated
* Knows if the file has been cleansed
* Knows if the file has been validated, and if so, whether it was successful or not
* Knows if the file has been ingested into the system database
* Knows if a user accepts the log file into the system in the event it’s invalid
* Can cleanse a textual log file
* Can validate a textual log file
* Can ingest a textual log file
* Knows all log files created.
* Can create a new log file
* Can modify an existing log file
* Can store a log file in system data store
* Can delete an existing log file
* Knows the line number where an error occurred in a log file
* Knows the error message explaining why a log file fails the validation test
* Can accept invalid log files as valid
* Can reject invalid log files
* Knows all enforcement action reports created
* Can create a new enforcement action report
* Can modify an existing enforcement action report
* Can store an enforcement action report in a data store
* Can delete an existing enforcement action report
* Knows the user-provided name of the vector
* Knows the user-provided description of the vector
* Knows the list of log entries that have been associated with it.
* Knows the list of nodes that make up the vector
* Can create a new node
* Can modify an existing node
* Can delete an existing node
* Knows the list of relationships between nodes in the vector.
* Can create a new relationship
* Can modify an existing relationship
* Can delete an existing relationship
* Can export the vector information in tabular format
* Knows the log entry number
* Knows the log entry timestamp
* Knows the log entry contents
* Knows the host IP address
* Knows the source log file
* Knows the source type
* Knows the node ID
* Knows the node name
* Knows the node timestamp
* Knows the node description
* Knows the log entry reference
* Knows the log entry information of the log entry reference
* Knows the node’s icon information
* Knows the node’s position on the graph
* Knows the visibility of the node’s ID on the graph
* Knows the visibility of the node’s name on the graph
* Knows the visibility of the node’s timestamp on the graph
* Knows the visibility of the node’s description on the graph
* Knows the visibility of the node’s log entry reference on the graph
* Knows the visibility of the node’s event type on a graph
* Knows the visibility of the node’s icon on a graph
* Knows the visibility of the node’s source on a graph
* Knows the node’s visibility on a graph
* Knows the icon name of the icon
* Knows the file path of the icon image
* Knows the list of nodes that make up the graph
* Knows the list of relationships that make up the graph
* Can export the nodes and relationships as an image.
* Can edit the positions of the nodes.
* Can associate nodes together as relationships
* Knows the file name of the file
* Knows the date and time it was most recently updated
* Knows if the file has been cleansed
* Knows if the file has been validated, and if so, whether it was successful or not
* Knows if the file has been ingested into the system database
* Knows if a user accepts the log file into the system in the event it’s invalid
* Can cleanse an audio log file
* Can validate an audio log file
* Can ingest an audio log file
* Knows the file name of the file
* Knows the date and time it was most recently updated
* Knows if the file has been cleansed
* Knows if the file has been validated, and if so, whether it was successful or not
* Knows if the file has been ingested into the system database
* Knows if a user accepts the log file into the system in the event it’s invalid
* Can cleanse a video log file
* Can validate a video log file
* Can ingest a video log file
* Knows the relationship ID of the relationship
* Knows the ID of the parent node of the relationship
* Knows the ID of the child node of the relationship
* Knows the relationship label
* Knows a list of vectors
* Can create a new vector
* Can modify an existing vector
* Can delete an existing vector
* Knows a list of vectors
* Can store vectors in a system datastore
* Can create a new vector
* Can modify an existing vector
* Can delete an existing vector
* Knows a list of pushed vector databases
* Can accept a pushed vector database
* Can reject a pushed vector database
* Can merge an accepted vector database
* Knows a list of vectors
* Can store vectors in a local datastore
* Can create a new vector
* Can modify an existing vector
* Can delete an existing vector
* Can push to the lead vector database
* Can pull from the lead vector database
* Knows the file name of the file
* Knows the date and time it was most recently updated
* Knows if the file has been cleansed
* Knows if the file has been validated, and if so, whether it was successful or not
* Knows if the file has been ingested into the system database
* Knows if a user accepts the log file into the system in the event it’s invalid
* Can cleanse an image log file
* Can validate an image log file
* Can ingest an image log file
* Knows the file name of the file
* Knows the date and time it was most recently updated
* Knows if the file has been cleansed
* Knows if the file has been validated, and if so, whether it was successful or not
* Knows if the file has been ingested into the system database
* Knows if a user accepts the log file into the system in the event it’s invalid
* Can cleanse a PDF log file
* Can validate a PDF image log file
* Can ingest a PDF image log file
* Knows a list of log entries ingested into the system.
* Can create a log entry
* Can delete an existing log entry
* Can store a log entry in a system data store
* Can edit an existing log entry
* Can search through log entries
* Can sort log entries
* Can associate a log entry to a vector
* Knows a list of all icons created
* Can create new icons
* Can delete an existing icon
* Can modify an existing icon
* Can store an icon in a system datastore
* Knows the IP address of the lead machine
* Knows the IP address of the server
* Knows the IP address of a user machine
* Can create a local network connection from the user machine to the lead machine
* Can create a local network connection from the lead machine to the user machine
* Can create a local network connection from the current machine to the server
* Can send data to the lead machine
* Can send data to the user machine
* Can send data to the server
* Can receive data from the server
* Can receive data from the lead machine
* Can receive data to the user machine
* Knows the IP address of the server
* Knows a list of all current server connections from client machines
* Can create a server port to accept connections
* Can send data to connected client machines
* Can receive data from connected client machines
* Can store received data to a system datastore
* Can retrieve data from a system datastore
* Can display event configuration information
* Can modify the information in an event configuration
* Can display Vector Manager information
* Can modify Vector Manager information
* Can display Log File Manager information
* Can display Enforcement Action Report Manager Information
* Can modify Enforcement Action Report Manager information
* Can display Log Entry Manager information
* Can modify Log Entry manager information
* Can display vector database information
* Can modify vector database information
* Can display Graph information
* Can modify Graph information
* Can display Icon Manager information
* Can modify Icon Manager information
* Can display Vector Information
* Can modify Vector Information

# CRC Cards

In this section, we describe the CRC cards of all the classes. A CRC card contains a class name, a class description, the superclasses of the class, the subclasses of a class, the responsibilities of the class, and the collaborations of the class.

## Event Configuration

|  |  |  |
| --- | --- | --- |
| Event Configuration | Superclass: N/A  Subclass: N/A  Description: A component of the system to be produced that will allow the analysts to define elements of the assessment. | Responsibilities:   1. Knows the name of the event 2. Knows the start timestamp of an event 3. Knows the end timestamp of an event 4. Knows the path to the root directory of an event 5. Knows the name of the red team folder 6. Knows the name of the blue team folder 7. Knows the name of the white team folder 8. Knows whether or not the current user is the lead 9. Knows the IP address of the lead 10. Knows the number of connections established to the lead 11. Can store all event configuration information in a system datastore 12. Can validate directory structure 13. Can validate directory existence 14. Can perform data cleansing 15. Can perform data ingestion   Collaborations:   1. Responsibilities 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 provide a service to Responsibility 1 from the Event Configuration Window class. 2. Responsibility 9 provides a service to Responsibility 1 from the Client Handler class 3. Responsibilities 2, 3, 4, 5, 6, 7 provide  service to Responsibility 2 from the Log File Manager class |

## Log File

|  |  |  |
| --- | --- | --- |
| Log File | Superclass: N/A  Subclass:   * Audio Log File * Video Log File * Image Log File * PDF Log File   Description: The source log files that the analysts will provide to the system to be cleansed and ingested. After that process, the output will be a series of log entries. | Responsibilities:   1. Knows the file name of the log file 2. Knows the date and time it was most recently updated 3. Knows if the file has been cleansed 4. Knows if the file has been validated, and if so, whether it was successful or not 5. Knows if the file has been ingested into the system database 6. Knows if a user accepts the log file into the system in the event it’s invalid 7. Can cleanse a textual log file 8. Can validate a textual log file 9. Can ingest a textual log file   Collaborations:   1. Responsibility 8 provides a service for Responsibility 2 from the Enforcement Action Report Manager Class 2. Responsibility 9 provides a service for Responsibility 2 from the Log Entry Manager class 3. This class is the superclass of Audio Log File, Video Log File, Image Log File, and PDF Log File |

## Log File Manager

|  |  |  |
| --- | --- | --- |
| Log File Manager | Superclass: N/A  Subclass: N/A  Description: A tool used to manage log files. | Responsibilities:   1. Knows all log files created. 2. Can create a new log file 3. Can modify an existing log file 4. Can store a log file in system data store 5. Can delete an existing log file   Collaborations:   1. Responsibility 1 provides a service to Responsibility 1 from the Log File Configuration 2. Responsibility 2 collaborates with Responsibility 3 from the Enforcement Action Report class 3. Responsibility 10 collaborates with Responsibility 9 from the Client Handler class |

## Enforcement Action Report

|  |  |  |
| --- | --- | --- |
| Enforcement Action Report | Superclass: N/A  Subclass: N/A  A report to be generated when a log file cannot be validated and shows any inaccurate data within that log file. | Responsibilities:   1. Knows the line number where an error occurred in a log file 2. Knows the error message explaining why a log file fails the validation test 3. Can accept invalid log files as valid 4. Can reject invalid log files   Collaborations:   1. Responsibility 2 collaborates with Responsibility 8 from the Log File Class 2. Responsibility 3 provides a service to Responsibility 2 from the Log File Manager class |

## Enforcement Action Report Manager

|  |  |  |
| --- | --- | --- |
| Enforcement Action Report Manager | Superclass: N/A  Subclass: N/A  Description: A tool used to manage enforcement action reports. | Responsibilities:   1. Knows all enforcement action reports created 2. Can create a new enforcement action report 3. Can modify an existing enforcement action report 4. Can store an enforcement action report in a data store 5. Can delete an existing enforcement action report   Collaborations:   1. Responsibility 1 provides a service for Responsibility 2 from the Log File Configuration class 2. Responsibilities 3, 4, 5 provide a service for Responsibility 3 from the Log File Configuration Class 3. Responsibility 4 collaborates with Responsibility 9 from the Client Handler class |

## Vector

|  |  |  |
| --- | --- | --- |
| Vector | Superclass: N/A  Subclass: N/A  Description: Created in the event configuration with just a name and a description, but after the analyst associates log entries to the vector and correlates nodes together, the collection of log entries in a vector can be used to describe some adversarial event or story between the blue and red team. | Responsibilities:   1. Knows the user-provided name of the vector 2. Knows the user-provided description of the vector 3. Knows the list of log entries that have been associated with it. 4. Knows the list of nodes that make up the vector 5. Can create a new node 6. Can modify an existing node 7. Can delete an existing node 8. Knows the list of relationships between nodes in the vector. 9. Can create a new relationship 10. Can modify an existing relationship 11. Can delete an existing relationship 12. Can export the vector information in tabular format   Collaborations:   1. Responsibility 4 provides a service for Responsibility 1 from the graph class 2. Responsibility 5 collaborates with Responsibility 8 from the Log Entry Manager class 3. Responsibility 8 provides a service to Responsibility 2 from the graph class 4. Responsibility 6 collaborates with Responsibility 4 from the Graph class 5. Responsibility 9 collaborates with Responsibility 5 from the Graph class 6. Responsibilities 1, 2, 3, 4, 8 provide a service to Responsibility 1 from the Vector Table Configuration class 7. Responsibilities 5, 6, 7, 9, 10, 11 collaborate with Responsibility 2 from the Vector Table Configuration class |

## Log Entry

|  |  |  |
| --- | --- | --- |
| Log Entry | Superclass: N/A  Subclass: N/A  Description: A record of an action made by a member of the red, white, or blue team. Log entries are the result of a log file being validated and ingested into the database within the system. | Responsibilities:   1. Knows the log entry number 2. Knows the log entry timestamp 3. Knows the log entry contents 4. Knows the host IP address 5. Knows the source log file 6. Knows the source type   Collaborations:   1. Responsibilities 1, 2, 3, 4, 5, 6 provide a service for Responsibility 6 in the Node class |

## Node

|  |  |  |
| --- | --- | --- |
| Node | Superclass: N/A  Subclass: N/A  Description: The visual representation of a particular significant log entry on the graph. Shown with an icon that should represent the type of event and is correlated with other nodes with connectors. | Responsibilities:   1. Knows the node ID 2. Knows the node name 3. Knows the node timestamp 4. Knows the node description 5. Knows the log entry reference 6. Knows the log entry information of the log entry reference 7. Knows the node’s icon information 8. Knows the node’s position on the graph 9. Knows the visibility of the node’s ID on the graph 10. Knows the visibility of the node’s name on the graph 11. Knows the visibility of the node’s timestamp on the graph 12. Knows the visibility of the node’s description on the graph 13. Knows the visibility of the node’s log entry reference on the graph 14. Knows the visibility of the node’s event type on a graph 15. Knows the visibility of the node’s icon on a graph 16. Knows the visibility of the node’s source on a graph 17. Knows the node’s visibility on a graph   Collaborations:   1. Responsibility 6 collaborates with Responsibilities 1, 2, 3, 4, 5, 6 in the Log Entry class 2. Responsibility 7 collaborates with Responsibilities 1, 2 in the Icon class 3. Responsibility 1 provides a service for Responsibilities 2, 3 in the Relationship class |

## Icon

|  |  |  |
| --- | --- | --- |
| Icon | Superclass: N/A  Subclass: N/A  Description: An icon is the visual representation of a node on a graph and is correlated with other nodes via arrow-head connectors. | Responsibilities:   1. Knows the icon name of the icon 2. Knows the file path of the icon image   Collaborations:   1. Responsibility 2 provides a service to Responsibility 7 from the Node Class |

## Graph

|  |  |  |
| --- | --- | --- |
| Graph | Superclass: N/A  Subclass: N/A  Description: The visual representation of a single given vector and its related nodes and connectors. | Responsibilities:   1. Knows the list of nodes that make up the graph 2. Knows the list of relationships that make up the graph 3. Can export the nodes and relationships as an image. 4. Can edit the positions of the nodes. 5. Can associate nodes together as relationships   Collaborations:   1. Responsibility 1 collaborates with Responsibility 4 from the Vector class 2. Responsibility 2 collaborates with Responsibility 8 from the Vector class 3. Responsibility 4 provides a service to Responsibility 6 from the Vector class 4. Responsibility 5 provides a service to Responsibility 9 from the Vector class 5. Responsibilities 1, 2 provide a service to Responsibility 1 from the Graph Configuration class 6. Responsibilities 4, 5 collaborates with Responsibility 2 from the Graph Configuration class |

## Audio Log File

|  |  |  |
| --- | --- | --- |
| Audio Log File | Superclass: Log File  Subclass: N/A  Description: A type of log file that will be transcribed into a log file to be cleansed and ingested. | Responsibilities:   1. Knows the file name of the file 2. Knows the date and time it was most recently updated 3. Knows if the file has been cleansed 4. Knows if the file has been validated, and if so, whether it was successful or not 5. Knows if the file has been ingested into the system database 6. Knows if a user accepts the log file into the system in the event it’s invalid 7. Can cleanse an audio log file 8. Can validate an audio log file 9. Can ingest an audio log file   Collaborations:   1. This class is a subclass of Log File and has the same collaborations it has |

## Video Log File

|  |  |  |
| --- | --- | --- |
| Video Log File | Superclass: Log File  Subclass: N/A  Description: A type of log file that will have its audio extracted and then transcribed into a log file to be cleansed, validated, and ingested. | Responsibilities:   1. Knows the file name of the file 2. Knows the date and time it was most recently updated 3. Knows if the file has been cleansed 4. Knows if the file has been validated, and if so, whether it was successful or not 5. Knows if the file has been ingested into the system database 6. Knows if a user accepts the log file into the system in the event it’s invalid 7. Can cleanse a video log file 8. Can validate a video log file 9. Can ingest a video log file   Collaborations:   1. This class is a subclass of Log File and has the same collaborations it has |

## Relationship

|  |  |  |
| --- | --- | --- |
| Relationship | Superclass: N/A  Subclass: N/A  Description: The association between a parent and child node within a vector. Is visually represented on a graph with a single arrow pointer. | Responsibilities:   1. Knows the relationship ID of the relationship 2. Knows the ID of the parent node of the relationship 3. Knows the ID of the child node of the relationship 4. Knows the relationship label   Collaborations:   1. Responsibilities 2, 3 collaborate with responsibility 1 of the Node class |

## Vector Database

|  |  |  |
| --- | --- | --- |
| Vector Database | Superclass: N/A  Subclass:   * Lead Vector Database * User Vector Database   Description: A database used to manage all known vectors. | Responsibilities:   1. Knows a list of vectors 2. Can create a new vector 3. Can modify an existing vector 4. Can delete an existing vector   Collaborations:   1. Responsibility 1 provides a service to Responsibility 1 from the Vector DB Configuration class 2. Responsibility 2, 3, 4 collaborate with Responsibility 2 from the Vector DB Configuration Class 3. This class is a superclass to Lead Vector Database and User Vector Database |

## Lead Vector Database

|  |  |  |
| --- | --- | --- |
| Lead Vector Database | Superclass: Vector Database  Subclass: N/A  Description: A database used to manage all known vectors. The lead will have the ability to accept or deny any changes to the vector database. Additionally, the lead vector database will have more access than the vector database. | Responsibilities:   1. Knows a list of vectors 2. Can store vectors in a system datastore 3. Can create a new vector 4. Can modify an existing vector 5. Can delete an existing vector 6. Knows a list of pushed vector databases 7. Can accept a pushed vector database 8. Can reject a pushed vector database 9. Can merge an accepted vector database   Collaborations:   1. This class is a subclass of Vector Database and has the same collaborations it has 2. Responsibility 6 collaborates with Responsibility 12 from the Client Handler class 3. Responsibility 6, 7 collaborate with Responsibility 8 from the Client Handler class 4. Responsibility 2 collaborates with Responsibility 9 from the Client Handler class |

## User Vector Database

|  |  |  |
| --- | --- | --- |
| User Vector Database | Superclass: Vector Database  Subclass: N/A  Description: A database used to manage all known vectors. A user will be able to make modifications to a vector and can push or pull from the lead vector database. | Responsibilities:   1. Knows a list of vectors 2. Can store vectors in a local datastore 3. Can create a new vector 4. Can modify an existing vector 5. Can delete an existing vector 6. Can push to the lead vector database 7. Can pull from the lead vector database   Collaborations:   1. This class is a subclass of Vector Database and has the same collaborations it has 2. Responsibility 6 collaborates with Responsibility 7 from the Client Handler class 3. Responsibility 7 collaborates with Responsibility 11 from the Client Handler class |

## Image Log File

|  |  |  |
| --- | --- | --- |
| Image Log File | Superclass: Log File  Subclass: N/A  Description: An image that will have its text extracted and stored within a log file to be cleansed, validated, and ingested. | Responsibilities:   1. Knows the file name of the file 2. Knows the date and time it was most recently updated 3. Knows if the file has been cleansed 4. Knows if the file has been validated, and if so, whether it was successful or not 5. Knows if the file has been ingested into the system database 6. Knows if a user accepts the log file into the system in the event it’s invalid 7. Can cleanse an image log file 8. Can validate an image log file 9. Can ingest an image log file   Collaborations:   1. This class is a subclass of Log File and has the same collaborations it has. |

## PDF Log File

|  |  |  |
| --- | --- | --- |
| PDF Log File | Superclass: Log File  Subclass: N/A  Description: A PDF that will be converted to a log file to be cleansed, validated, and ingested. | Responsibilities:   1. Knows the file name of the file 2. Knows the date and time it was most recently updated 3. Knows if the file has been cleansed 4. Knows if the file has been validated, and if so, whether it was successful or not 5. Knows if the file has been ingested into the system database 6. Knows if a user accepts the log file into the system in the event it’s invalid 7. Can cleanse a PDF log file 8. Can validate a PDF image log file 9. Can ingest a PDF image log file   Collaborations:   1. This class is a subclass of Log File and has the same collaborations it has |

## Log Entry Manager

|  |  |  |
| --- | --- | --- |
| Log Entry Manager | Superclass: N/A  Subclass: N/A  Description: A tool used to manage log entries. | Responsibilities:   1. Knows a list of log entries ingested into the system. 2. Can create a log entry 3. Can delete an existing log entry 4. Can store a log entry in a system data store 5. Can edit an existing log entry 6. Can search through log entries 7. Can sort log entries 8. Can associate a log entry to a vector   Collaborations:   1. Responsibility 2 collaborates with Responsibility 9 from, the Log File class 2. Responsibility 8 provides a service for Responsibility 5 from the Vector class 3. Responsibility 1 provides a service for Responsibility 1 from the Log Entry Configuration class 4. Responsibilities 5, 6, 7, 8 collaborate with Responsibility 2 from the Log Entry Configuration class 5. Responsibility 4 collaborates with Responsibility 9 from the Client Handler class |

## Icon Manager

|  |  |  |
| --- | --- | --- |
| Icon Manager | Superclass: N/A  Subclass: N/A  Description: A tool used to manage icons. | Responsibilities:   1. Knows a list of all icons created 2. Can create new icons 3. Can delete an existing icon 4. Can modify an existing icon 5. Can store an icon in a system datastore   Collaborations:   1. Responsibility 1 provides a service for Responsibility 1 from the Icon Configuration class 2. Responsibility 2, 3, 4 collaborate with Responsibility 2 from the Icon Configuration class 3. Responsibility 5 collaborates with Responsibility 9 from the Client Handler class |

## Client Handler

|  |  |  |
| --- | --- | --- |
| Client Handler | Superclass: N/A  Subclass: N/A  A tool sitting on the server side that handles the communication between the server and the client. | Responsibilities:   1. Knows the IP address of the lead machine 2. Knows the IP address of the server 3. Knows the IP address of a user machine 4. Can create a local network connection from the user machine to the lead machine 5. Can create a local network connection from the lead machine to the user machine 6. Can create a local network connection from the current machine to the server 7. Can send data to the lead machine 8. Can send data to the user machine 9. Can send data to the server 10. Can receive data from the server 11. Can receive data from the lead machine 12. Can receive data to the user machine   Collaborations:   1. Responsibility 9 provides a service to Responsibility 5 from the Server Handler class 2. Responsibility 1 collaborates with Responsibility 9 from the Event Configuration class 3. Responsibility 12 provides a service to Responsibility 6 from the Lead Vector Database class 4. Responsibility 8 provides a service for Responsibilities 6, 7 collaborates from the Lead Vector Database class 5. Responsibility 7 provides a service for Responsibility 6 from the User Vector Database class 6. Responsibility 11 provides a service to Responsibility 7 from the User Vector Database class 7. Responsibility 10 collaborates with Responsibility 4 from the Server Handler class 8. Responsibility 9 provides a service for Responsibility 5 from the Icon Manager class 9. Responsibility 9 provides a service for Responsibility 4 from the Log Entry Manager class 10. Responsibility 9 provides a service for Responsibility 2 from the Lead Vector Database class 11. Responsibility 9 provides a service for Responsibility 4 from the Enforcement Action Report Manager class 12. Responsibility 9 provides a service for Responsibility 10 from the Log File Manager class |

## Server Handler

|  |  |  |
| --- | --- | --- |
| Server Handler | Superclass: N/A  Subclass: N/A  Description: A tool sitting on the client side that handles the communication between the server and the client. | Responsibilities:   1. Knows the IP address of the server 2. Knows a list of all current server connections from client machines 3. Can create a server port to accept connections 4. Can send data to connected client machines 5. Can receive data from connected client machines 6. Can store received data to a system datastore 7. Can retrieve data from a system datastore   Collaborations:   1. Responsibility 5 collaborates with Responsibility 9 from the Client Handler class 2. Responsibility 4 provides a service to Responsibility 10 from the Client Handler class |

## Event Configuration Window

|  |  |  |
| --- | --- | --- |
| Event Configuration Window | Superclass: N/A  Subclass: N/A  Description: Allows you to manage an event configuration. | Responsibilities:   1. Can display event configuration information 2. Can modify the information in an event configuration   Collaborations:   1. Responsibility 1 collaborates with Responsibilities 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 from the Icon class |

## Vector Configuration

|  |  |  |
| --- | --- | --- |
| Vector Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage a vector configuration. | Responsibilities:   1. Can display Vector Manager information 2. Can modify Vector Manager information   Collaborations:   1. Responsibility 1 collaborates with Responsibility 1 from the Vector class 2. Responsibility 2 provides a service to Responsibility 2, 3, 4 from the Vector class |

## Log File Configuration

|  |  |  |
| --- | --- | --- |
| Log File Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage a log file configuration. | Responsibilities:   1. Can display Log File Manager information 2. Can display Enforcement Action Report Manager Information 3. Can modify Enforcement Action Report Manager information   Collaborations:   1. Responsibility 1 collaborates with Responsibility 1 from the Log File Manager 2. Responsibility 2 collaborates with Responsibility 1 from the Enforcement Action Report Manager Class 3. Responsibility 3 collaborates with Responsibilities 3, 4, 5 from the Enforcement Action Report Manager class |

## Log Entry Configuration

|  |  |  |
| --- | --- | --- |
| Log Entry Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage a log entry configuration. | Responsibilities:   1. Can display Log Entry Manager information 2. Can modify Log Entry manager information   Collaborations:   1. Responsibility 1collaborates with Responsibility 1 from the Log Entry Manager class 2. Responsibility 2 provides a service for Responsibilities 5, 6, 7, 8 from the Log Entry Manager class |

## Vector Db Configuration

|  |  |  |
| --- | --- | --- |
| Vector DB Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage a vector database configuration. | Responsibilities:   1. Can display vector database information 2. Can modify vector database information   Collaborations:   1. Responsibility 1 collaborates with Responsibility 1 from the Vector Database class 2. Responsibility 2 provides a service to Responsibilities 2, 3, 4 from the Vector Database class |

## Graph Configuration

|  |  |  |
| --- | --- | --- |
| Graph Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage a graph configuration. | Responsibilities:   1. Can display Graph information 2. Can modify Graph information   Collaborations:   1. Responsibility 1 collaborates with Responsibilities 1, 2 from the Graph class 2. Responsibilities 2 provides a service to Responsibilities 4, 5 from the Graph |

## Icon Configuration

|  |  |  |
| --- | --- | --- |
| Icon Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage an icon configuration. | Responsibilities:   1. Can display Icon Manager information 2. Can modify Icon Manager information   Collaborations:   1. Responsibility 1 provides a service for Responsibility 1 from the Icon Configuration class 2. Responsibility 2 provides a service for Responsibilities 2, 3, 4 from the Icon Manager class |

## Vector Table Configuration

|  |  |  |
| --- | --- | --- |
| Vector Table Configuration | Superclass: N/A  Subclass: N/A  Description: Allows you to manage a vector table configuration. | Responsibilities:   1. Can display Vector Information 2. Can modify Vector Information   Collaborations:   1. Responsibility 1 collaborates with Responsibilities 1, 2, 3, 4, 8 from the Vector class 2. Responsibility 2 collaborates with Responsibilities 5, 6, 7, 9, 10, 11 collaborate from the Vector class |